

Guidelines for Adopting Behavior-Driven Development (BDD): A Case Study

Shexmo Richarlison Ribeiro dos Santos¹^a, Marcus Vinicius Santana Silva¹^b,
Marcos Cesar Barbosa dos Santos¹^c, Luiz Felipe Cirqueira dos Santos¹^d,
Mariano Florencio Mendonça¹^e, Marcos Venicius Santos¹^f, Marckson Fábio da Silva Santos¹^g,
Alberto Luciano de Souza Bastos¹^h and Fabio Gomes Rocha²ⁱ

¹ Federal University of Sergipe, São Cristóvão, Sergipe, Brazil

² ISH (SafeLabs), Vitória, Espírito Santo, Brazil

Keywords: Behavior-Driven Development (BDD), Case Study, Guidelines, Survey.

Abstract: **Context:** To analyze the effectiveness of adopting Behavior-Driven Development (BDD) with agile teams that have not previously used this framework. **Problem:** Lack of a study that combines guidelines inherent to adopting BDD. **Solution:** To achieve the proposed objective, the authors carried out two previous studies: the first identifying the state of the art of BDD through a Systematic Multivocal Literature Review; and, consequently, the state of the practice was analyzed, through a Survey, with professionals who use BDD in their work activities. Thus, this research sought to validate the guidelines found. **Method:** A case study was carried out in a private company focused on security software development with an agile team that did not use BDD. **Summarization of results:** The results obtained showed the effectiveness and validity of the guidelines previously found for adopting BDD. There is also a need for improving communication among stakeholders, the only problem found in BDD's adoption. **Contributions and impact:** The main contribution of this study is the guidelines presented inherent to the adoption of BDD to contribute to both academia and industry regarding the understanding and implementation of this framework, respectively.

1 INTRODUCTION

The Software Development Life Cycle is made up of all the stages inherent to its construction (Rajbhoj et al., 2024). Requirements Engineering is the part of software construction that aims to make it work as expected through frameworks and tools. Frameworks are understood as models used in a systemic and standardized manner to achieve an objective.

Behavior-Driven Development (BDD) is an agile

framework used in the software life cycle and has gained notoriety among researchers and professionals. BDD applies to all stages of software development, from requirements elicitation to code documentation. Because it has a ubiquitous language that is easy to understand by all parties, BDD focuses on the behavior the software is expected to perform to be more assertive in eliciting requirements (Pereira et al., 2018).

Therefore, a case study was conducted to evaluate the possibility of applying guidelines to adopting BDD and to validate previous studies regarding this framework. Part of the Goal-Question-Metric protocol (Wohlin et al., 2012) was used, namely: “**To analyze the adoption of BDD, with the purpose of implementing, in relation to agile teams, from the point of view of professionals, in the context of software development**”. To achieve this goal, the research questions are presented in Table 1.

To answer the research questions and achieve the

^a <https://orcid.org/0000-0003-0287-8055>

^b <https://orcid.org/0009-0000-9211-5259>

^c <https://orcid.org/0000-0002-7929-3904>

^d <https://orcid.org/0000-0003-4538-5410>

^e <https://orcid.org/0000-0003-0732-3980>

^f <https://orcid.org/0009-0006-1645-6127>

^g <https://orcid.org/0009-0001-6479-1900>

^h <https://orcid.org/0009-0002-3911-9757>

ⁱ <https://orcid.org/0000-0002-0512-5406>

Table 1: Research questions.

Question	Definition	Justification
Q1	What are the guidelines inherent to adopting BDD?	Present the guidelines to facilitate the adoption of BDD.
Q2	Did BDD achieve the expected purpose concerning software development?	Identify whether BDD applies to teams that have never used it to develop software.
Q3	What is the respondents perception that adopted BDD for this case study?	Present positive and negative points through a survey regarding the adoption of BDD.
Q4	Are the guidelines presented effective for adopting BDD?	Validate the characteristics found by the authors in previous studies.

general objective of this study, a case study and a survey were conducted with a team that did not use BDD in its work activities at a private Brazilian security software development company.

2 BEHAVIOR-DRIVEN DEVELOPMENT (BDD)

Frameworks are structures designed to help automate software development tasks to save time and reduce costs. Behavior-Driven Development (BDD) is a framework used throughout the software life cycle, from requirements elicitation through documentation to software maintenance. Created in 2003 by Dan North (North, 2006), BDD aims to mitigate the problems encountered in Test-Driven Development (TDD). While the latter focuses on testing the requirements themselves, BDD aims to be written according to the expected behavior of the software.

BDD is written using ubiquitous language, i. e., in everyday natural language, to obtain clear and concise communication regarding the expected objectives. The structure used is through the commands “**given**, **when** and **then**”, a pattern called *gherkin*, so that each user story must follow it. In this way, BDD contributes to assertive deliveries about requirements elicitation (Pereira et al., 2018), generating time savings and less rework for the team.

Given the lack of scientific research with such guidelines, the authors previously carried out a Sys-

tematic Multivocal Literature Review¹, and a Survey². The authors yet carried out two more additional studies: a Case Study³, and an Experiment⁴ to better understand the BDD framework. Thus, the results found were validated through this current case study.

3 RELATED WORK

Binamungo and Maro (Binamungo and Maro, 2023) systematically mapped BDD so that they could contribute with aspects related to the state of the art in which BDD is found. The authors were able to observe the need for more empirical evidence. Thus, this case study may contribute as it is a real study with an industry team and presents guidelines aimed at adopting BDD.

Kudo et al. (Kudo et al., 2023) carried out a study to align requirements and tests through patterns aiming at software quality. The authors created a framework called Software Pattern Meta-Model (SoPaMM) to align requirements and tests, providing testable patterns and reducing cost and project development time. Thus, the authors pointed out the need to create testable requirements focusing on the behavior expected by the software through reusable patterns using the SoPaMM framework. Therefore, it is interesting to use well-defined standards to guarantee the quality of the final product delivered, enabling the meeting of what is expected from the product with the stakeholder’s desire.

Santos et al. (Santos et al., 2024) carried out a case study using BDD to elicit non-functional requirements following the ISO/IEC/IEEE 25010 Standard. The authors reported the effectiveness of BDD in eliciting non-functional requirements since they are more complex than functional requirements. Thus, this new case study may contribute to demonstrating the efficacy of using BDD with agile teams that do not use it and may prove its ease of adoption in a real context.

4 METHODOLOGY

4.1 Case Study

The research was conducted in a private Brazilian company operating in the cybersecurity software development sector. It involved four development

¹<https://encurtador.com.br/4qBWg>.

²<https://encurtador.com.br/r6qha>.

³<https://www.mdpi.com/2674-113X/3/3/14>.

⁴This study is under review in a Journal.

teams, totaling 56 participants. Each team consisted of approximately 10 to 15 members, with a designated team leader responsible for consolidating the acquired knowledge and representing the team in the final evaluations.

The study began with a presentation by one of the researchers introducing the BDD adoption guidelines. This presentation covered the BDD development cycle, its key concepts, best practices, and the use of prompts for large language models (LLMs) and related tools. Following this introduction, participants engaged in a hands-on workshop, allowing them to experiment directly with the proposed approach and tools.

After the introductory phase, teams applied the guidelines and tools over six sprints, each lasting 15 days. The researchers remained available throughout the study to clarify doubts and provide technical support; however, no direct intervention was necessary. At the end of the study period, team leaders completed a structured questionnaire to evaluate the applicability and effectiveness of the adopted guidelines.

4.2 Survey

A survey was conducted using structured questionnaires to collect insights and validate the adoption of the BDD guidelines. This method helps understand group behaviour and experiences in a given context and is widely employed in empirical studies due to its ease of application and low operational cost (Punter et al., 2003).

The questionnaire consisted of 13 closed-ended questions, structured as described in Table 2 and was designed to gather data on the teams' experiences throughout the six sprints, addressing aspects such as the clarity of the guidelines, the usefulness of LLM prompts, the impact on development quality, and the challenges encountered during BDD adoption. The research followed an exploratory approach (Wohlin et al., 2006), aiming to identify relevant information that could contribute to refining the proposed guidelines and improving their applicability in agile teams.

5 RESULTS

Through the studies carried out by the authors mentioned in Section 2, it was possible to outline guidelines for the adoption of Behavior-Driven Development (BDD). The guidelines were summarized in Figure 1.

Regarding the human/behavioral aspect, it is possible to observe 5 points:

- Adoption of BDD for functional and non-functional requirements: since BDD is a framework used throughout the software life cycle, it is possible to adopt it both for eliciting functional and non-functional requirements;
- 3 amigos meeting: initial meeting held by the product owner, tester, and developer to idealize what is expected of the functionalities inherent to the software development;
- Scenario review: necessary review inherent to the software development process to see if what is being developed is following what is expected of the final product;
- Standardization of communication between stakeholders: use of everyday language without the need for jargon or technical terms;
- Adoption of writing standards: to maintain everyone's understanding of what is being discussed.

Regarding the technological aspects, 3 points are presented:

- Adoption of BDD with the Gherkin standard: to contribute to the standardization of the steps inherent to the use of BDD, that is, the terms "given, when, then" used in the requirements elicitation stage;
- Use of LLMs to support writing and automation: BDD can contribute to the refinement of the prompt to bring quality to the delivery of the output performed by the LLM. Part of the software development process was automated, an aspect that contributed to the adoption of BDD;
- Living documentation: documentation of the code performed concomitantly with its development so that it is not a stage performed only when the final product is delivered.

Figure 1 presents 13 points which were used for serving as the basis for the questionnaire applied to the target team of this study. Since the team had not used BDD before, creating a script to understand the topics presented in the questions was necessary. The script is expressed in a Figure which can be seen here⁵ and presents the components that it indicates as good practices for adopting BDD. The script guided respondents with basic terms in the context of BDD, such as gherkin language.

⁵<https://encurtador.com.br/DORLK>

Table 2: Survey Questions.

Question	Definition	Justification
SQ1	Did BDD achieve the team's goal?	Evaluate BDD effectiveness.
SQ2	How easy was it to adopt BDD in development?	Assess understanding and learning curve.
SQ3	Is BDD useful as technological support?	Analyze supportiveness in development.
SQ4	Is the Gherkin pattern effective with BDD?	Evaluate Gherkin clarity and usage.
SQ5	How useful is LLM for writing BDD scenarios?	Assess writing support using LLMs.
SQ6	How useful is LLM for automating tests in BDD?	Analyze LLM impact on test automation.
SQ7	Does BDD enhance living documentation?	Emphasize its role in up-to-date docs.
SQ8	Does BDD improve human interactions?	Explore human-centered aspects.
SQ9	Is BDD effective for requirement elicitation?	Assess BDD in gathering requirements.
SQ10	Does BDD aid in "3 amigos" meetings?	Analyze alignment and collaboration.
SQ11	Is BDD helpful in scenario reviews?	Examine ease of scenario refinement.
SQ12	Does BDD improve stakeholder communication?	Discuss communication facilitation.
SQ13	Does BDD promote standardized writing?	Evaluate consistency in expressions.

5.1 Survey Results

5.1.1 SQ1 - Did BDD Manage to Achieve the Objective Expected by the Team?

The responses obtained were 50% for **yes**, 50% for **partially**, and no response for **no**. One of the respondents added the following comment on the use of BDD.

BDD was an effective tool for achieving the team's technical goals, such as increasing clarity in requirements and promoting the creation of automated tests based on use cases. However, a significant part of the goals were not achieved, and this was not due to the BDD methodology but rather to the lack of integration and collaboration between the development team and stakeholders. The lack of consistent alignment between both parties resulted in a lack of communication and compromised delivery of some expected benefits. Therefore, the problem is not with BDD but stakeholders' lack of support and active involvement throughout the process.

The adoption of BDD itself is not a problem for teams that do not use it, but rather the collaboration between stakeholders so that it can directly affect the quality of the final product. Through the script presented before, it is possible, in theory, to achieve the expected objective from the software's behavior.

From questions SQ2 to SQ13, the response options, following the Linkert scale, were very bad, bad, good, very good, and excellent.

5.1.2 SQ2 - What Is Your Perception Regarding the Adoption of BDD for Software Development?

The responses obtained were 50% for **excellent**, and 50% for **very good**. From the point of view of the 4 team leaders, the adoption of BDD occurs satisfactorily in the context of software development.

5.1.3 SQ3 - What Is Your Perception Regarding the Adoption of BDD as Technological Support?

The responses obtained were 50% for **very good**, and 50% for **good**. We were able to ingest positive aspects in the adoption of BDD as technical support, given its adoption in a succinct and easy-to-understand manner among stakeholders.

5.1.4 SQ4 - What Is Your Perception Regarding the Adoption of BDD with the Gherkin Pattern?

The responses obtained were 50% for **very good**, 25% for **excellent**, and 25% for **good**. Using the Gherkin pattern, it is possible to achieve the expected behavior of the software more directly than with traditional frameworks. Thus, using the Gherkin pattern can bring quality to software development and save time since rework can be reduced, and the time spent redoing one step of the process can be used in other steps.

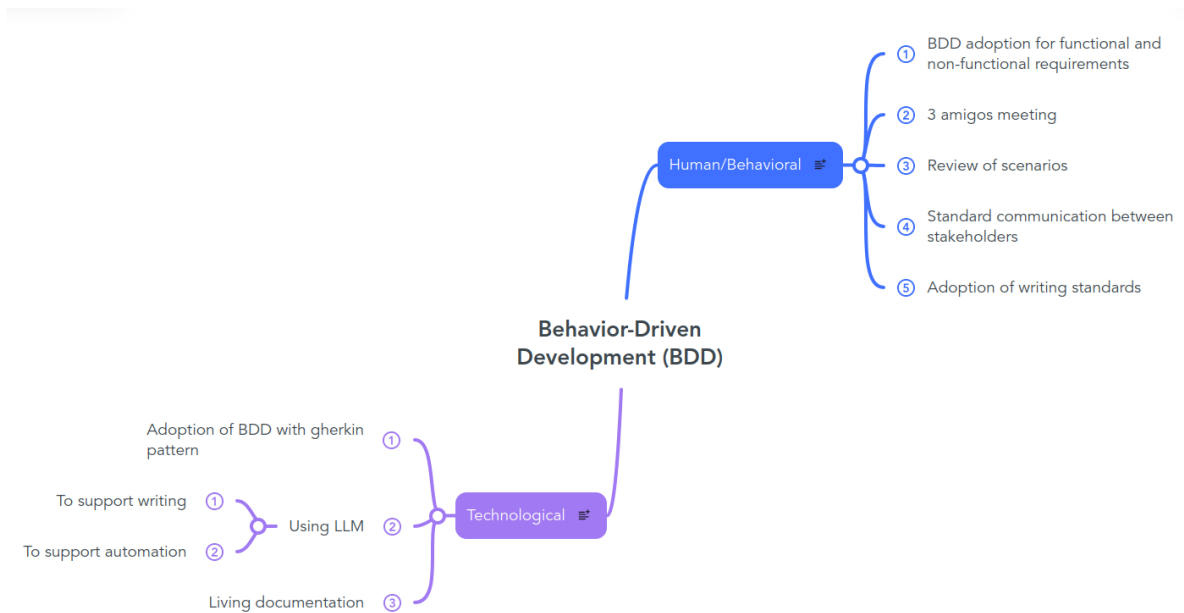


Figure 1: Mind map for BDD adoption.

5.1.5 SQ5 - What Is Your Perception Regarding the Adoption of BDD Using Large-Language Models (LLM) to Support Writing?

The responses obtained were 50% for **very good**, and 50% for **good**. Using prompts written in a way that is appropriate to the desired behavior of the software, BDD can contribute to requirements elicitation and test automation. As shown in the Figure 1, BDD can enhance its capabilities using LLMs when used correctly.

5.1.6 SQ6 - What Is Your Perception Regarding the Adoption of BDD Using LLM to Support Test Automation?

The responses obtained were 50% for **very good**, and 50% for **good**. Using BDD together with LLMs can contribute to the quality of the final product since the monotonous work of requirements elicitation, for example, can be automated through new or reused prompts for specific software behaviors.

5.1.7 SQ7 - What Is Your Perception Regarding the Adoption of BDD in Relation to Living Documentation?

The responses obtained were 50% for **very good**, 25% for **excellent**, and 25% **good**. Software documentation is an essential step, as any other step, in developing software, although it is often neglected. By having documentation carried out simultaneously with software development, there are benefits such as

generating savings if it is necessary to return the software to a certain point where there was no bug, for example, because with documentation, it is known precisely what, why and how a particular step in the process was done.

5.1.8 SQ8 - What Is Your Perception Regarding the Adoption of BDD in Human/Behavioral Relationships?

The responses obtained were 50% for **good**, 25% for **very good**, and 25% **bad**. The need for stakeholder participation in software development is essential. For software to be developed correctly and to meet stakeholder requirements, stakeholders must be present at meetings required during the development process to validate whether what is being done is what stakeholders expect. When stakeholders neglect development, the possibility of delivering unsatisfactory software is greater than if they had participated during the process.

5.1.9 SQ9 - What Is Your Perception Regarding the Adoption of BDD to Elicit Functional and Non-Functional Requirements?

The responses obtained were 50% for **very good**, 25% for **excellent**, and 25% **good**. Non-functional requirements are more complex than functional requirements. The requirements elicitation stage needs to be well defined so that the software achieves the behaviors expected by stakeholders. In the elicitation of non-functional requirements, it is necessary to have clarity and conciseness in the requirements, given the

need to translate them into testable scenarios. The gherkin language of BDD contributes to the required conciseness inherent in eliciting non-functional requirements.

5.1.10 SQ10 - What Is Your Perception Regarding the Adoption of BDD for the “3 Amigos” Meeting?

The responses obtained were 75% for **good**, and 25% for **bad**. The “3 friends” meeting was one of the guidelines we identified for adopting BDD. Although it is an essential step for its execution, it is vital that the design made by the product owner, developers, and testers is harmonious since if the meeting is not well done, this will reflect negatively on the subsequent stages of software development, and features may be built for pains that do not exist.

5.1.11 SQ11 - What Is Your Perception Regarding the Adoption of BDD on Scenarios Review?

The responses obtained were 25% for **excellent**, 25% for **very good**, 25% for **good**, and 25% for **bad**. BDD uses user stories created with testable scenarios once requirements that can be implemented in the software are elicited. Reviewing these scenarios is objective to the proposed requirement since the Gherkin language is done directly. It is important to emphasize the need for adequate writing for the elicitation of requirements since the review of scenarios is an additional step to guarantee the final product's quality, and it is not necessary to carry out the requirements elicitation step in its entirety once again.

5.1.12 SQ12 - What Is Your Perception Regarding the Adoption of BDD in Relation to Communication Between Stakeholders?

The responses obtained were 50% for **good**, 25% for **excellent**, and 25% for **bad**. For the final product to be high quality, all those involved must actively participate in developing the software. Suppose some parts do not understand their role in the process. In that case, the software will likely fail to achieve satisfactory quality since each stakeholder plays a different role. The communication proposed by BDD is that the process be carried out clearly with all stakeholders. If communication is flawed or presents ambiguities, the adoption of BDD is not being done correctly.

5.1.13 SQ13 - What Is Your Perception Regarding the Adoption of BDD in Relation to Standardized Writing?

The responses obtained were 75% for **excellent**, and 25% for **very good**. Standardized writing facilitates stakeholders' understanding during the software development process. The use of jargon, for example, can make it difficult for stakeholders from different fields to understand. BDD must be adopted using everyday language, without technical terms, so everyone can understand what a given part of the software needs to achieve, for example. Using the “given, when, then” pattern helps to assertively deliver the requirement so that it is outlined within a testable scenario.

5.2 Research Questions

5.2.1 Q1 - What Are the Guidelines Inherent to Adopting BDD?

The guidelines for adopting BDD are presented in Figure 1. These guidelines were outlined through a Systematic Multivocal Literature Review, a Case Study, a Survey, and an Experiment. The adoption of BDD can contribute to the quality of the software as long as it is carried out correctly and follows the guidelines presented. This work can guide companies that are not using BDD in their work activities to adopt it in the software development process.

5.2.2 Q2 - Did BDD Achieve the Expected Purpose Concerning Software Development?

As we can see in the survey results, BDD presents positive aspects in all steps of its implementation, seen by the team leaders as a framework that promotes quality for the final product. The point that all four respondents mentioned as unfavorable concerning the adoption of BDD was regarding the interaction between stakeholders, so the problem lies in something other than BDD itself, but rather in the communication between the partakers.

5.2.3 Q3 - What Is the Respondents Perception that Adopted BDD for this Case Study?

The survey showed positive results in all questions except for specific responses related to interpersonal relationships and scenarios review. Suppose BDD is not transparent to all stakeholders. In that case, the probability of the software being developed in a way that is not expected is greater than when the functionalities

have precise purposes. An additional comment was made by one of the participants regarding the Meeting of the “3 Amigos”.

We noticed a lack of genuine collaboration in the “3 Amigos” meeting, which was supposed to facilitate alignment between developers, testers, and business stakeholders. Although the physical presence of participants was guaranteed, active participation and engagement, especially from a business perspective, were insufficient. This created gaps in understanding between the development team and stakeholders, leaving some issues open and compromising the clarity needed for agile and well-directed development. In this format, the meeting could benefit from more detailed preparation and more proactive involvement of business representatives.

Once again, the lack of collaboration from one of the parties was mentioned negatively concerning the adoption of BDD. Suppose the meeting of the “3 amigos” is not outlined correctly. In that case, there will likely be problems in the behaviors developed for the software since the functionalities were not elicited objectively.

Therefore, the perception of team leaders regarding adopting BDD is positive. The problem mentioned only occurs concerning the communications required during the process.

5.2.4 Q4 - Are the Guidelines Presented Effective for Adopting BDD?

Yes, the guidelines presented are effective. Adopting BDD has proven effective for teams that are not using it in their software development activities. Because it has an everyday language, it has shown that when used correctly, as shown in the Figure, BDD contributes to the quality of the delivery of the final product.

The crucial point raised by all respondents was the lack of communication between stakeholders. One of the respondents made an additional comment.

BDD provides a good basis for improving communication between stakeholders, as it transforms business requirements into language accessible to both the development team and those involved. However, we realized that the process could be optimized using visual tools and a more interactive approach, simplifying communication and maintaining stakeholders’ interest over time. For example, adopting diagrams or visual dashboards could facilitate the understanding of

usage scenarios and promote more fluid and efficient interaction between stakeholders and the development team.

It is interesting to highlight the recommendation of one of the respondents regarding using visual media for better collaboration among stakeholders, given the perceived failure in communication. The respondents viewed the guidelines positively, so the main obstacle was how stakeholders neglected communication during software development.

6 DISCUSSIONS

As a framework used throughout the software life cycle, BDD can go through the steps of the software development process, from elicitation to documentation of requirements. It is important to realize how complete BDD is, with a standardized language that helps both in testing and in the validity and verification of features, assuming an essential role in software quality.

Just Santos et al. (Santos et al., 2024), we were able to verify the effectiveness of BDD through a case study, in this case, the effectiveness of the guidelines that involve its adoption. There is a need to improve communication between stakeholders so that BDD can present the maximum positive results.

Binamungo and Maro (Binamungo and Maro, 2023) sought to analyze the state of the art that BDD was in, while Kudo et al. (Kudo et al., 2023) showed the need for aligned writing standards for requirements and tests. Our study was able to advance the understanding of BDD regarding the visual need that can be implemented concerning stakeholders, as well as showing the benefits of standardized writing that BDD has.

Through the responses obtained, we observed that the only point to be improved does not concern BDD, and all four respondents stated this, but the need for collaboration on the part of stakeholders. There is no point in the team being competent in its activities if no stakeholders can validate what is being developed, so the chance of functionalities being developed correctly, but for problems that do not exist is high.

It is essential to highlight that the only negative point regarding adopting BDD is not directly linked to the framework but to the stakeholders who leave the development process at the mercy of the technical team. For BDD to be used with quality, all parties involved in the process must be aligned so that everyone commonly understands the objective. The suggestion of one of the respondents who mentioned using dashboards could be beneficial in this context, as it helps

stakeholders understand.

7 CONCLUSION

The main point we observed through the survey used in this case study was the perception that when adopting BDD for software development, the point that needs to be improved is due to the commitment of stakeholders, given the need to validate what the team has done. All parties involved in software development are essential, as each one plays a different role. If there is no client to verify the features being developed, the features will likely end up achieving purposes that do not exist.

The main contribution of this study is to show that even though BDD presents itself as a framework for effective use in software development if there is inadequate communication between the team and external stakeholders, the delivery of the final product will suffer negative consequences. Furthermore, the recommendation of one of the respondents to use diagrams and dashboards to help visualize external stakeholders is a crucial point to be mentioned, as it can lead to a significant improvement in the communication inherent to the process.

As suggestions for future work, an experiment could be carried out with teams that use BDD, using diagrams and dashboards with a focus on understanding and interacting with external stakeholders. This way, we can analyze whether the use of these resources present improvements to the development process of the software.

ACKNOWLEDGEMENTS

Shexmo Santos would like to thank CAPES for the financial support (Bolsa de mestrado process no. 88887.888613/2023-00).

REFERENCES

- Binamungu, L. P. and Maro, S. (2023). Behaviour driven development: A systematic mapping study. *Journal of Systems and Software*, 203:111749.
- Kudo, T. N., Bulcão-Neto, R. d. F., Neto, V. V. G., and Vincenzi, A. M. R. (2023). Aligning requirements and testing through metamodeling and patterns: design and evaluation. *Requirements Engineering*, 28(1):97–115.
- North, D. (2006). Introducing BDD. <https://dannorth.net/introducing-bdd/>.
- Pereira, L., Sharp, H., de Souza, C., Oliveira, G., Marczak, S., and Bastos, R. (2018). Behavior-Driven Development benefits and challenges: reports from an industrial study. In *Proceedings of the 19th International Conference on Agile Software Development: Companion*, pages 1–4.
- Punter, T., Ciolkowski, M., Freimut, B., and John, I. (2003). Conducting on-line surveys in software engineering. In *2003 International Symposium on Empirical Software Engineering, 2003. ISESE 2003. Proceedings.*, pages 80–88. IEEE.
- Rajbhoj, A., Somase, A., Kulkarni, P., and Kulkarni, V. (2024). Accelerating software development using generative ai: Chatgpt case study. In *Proceedings of the 17th Innovations in Software Engineering Conference, ISEC '24*, New York, NY, USA. Association for Computing Machinery.
- Santos, S., Pimentel, T., Rocha, F. G., and Soares, M. S. (2024). Using behavior-driven development (bdd) for non-functional requirements. *Software*, 3(3):271–283.
- Wohlin, C., Höst, M., and Henningsson, K. (2006). Empirical research methods in web and software engineering. *Web engineering*, pages 409–430.
- Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., and Wesslén, A. (2012). *Experimentation in software engineering*. Springer Science & Business Media.